CMCS Notification

Karen Schuchardt

karen.schuchardt@pnl.gov

Pacific Northwest National Laboratory

June 2002

CMCS Notification

Goal: Provide (build and acquire) messaging technology needed by the CMCS adaptive architecture and create add-on messaging services that support loose coordination between CMCS users and applications.

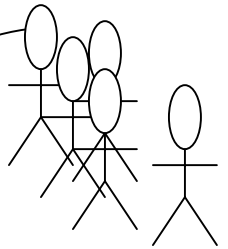
Team: Karen Schuchardt (lead), Brett Didier, Carina Lansing, Jim Myers, Carmen Pancerella

Why use messaging?

- You can request notifications on activities/data in your research area. These notifications can come to you as a user or be incorporated into your application, enabling event-based workflow.
  - Tell me when information about molecular species xyz has been added
  - Tell me when files of type abc are created or modified
  - Tell me when reviewers comment on my data
- Your applications can react to and send events regardless of the source or destination. This allows loose coordination between applications and helps reduce the drawbacks of designing large monolithic system.

Use Case – Notification and CMCS Data Services

Get,put document



Pedigree,
Annotations…

Put,get document,
Add/change metadata
(all DAV operations)

CMCS
Data
Server(s)
(SAM)

Messaging
Service

Data Miner

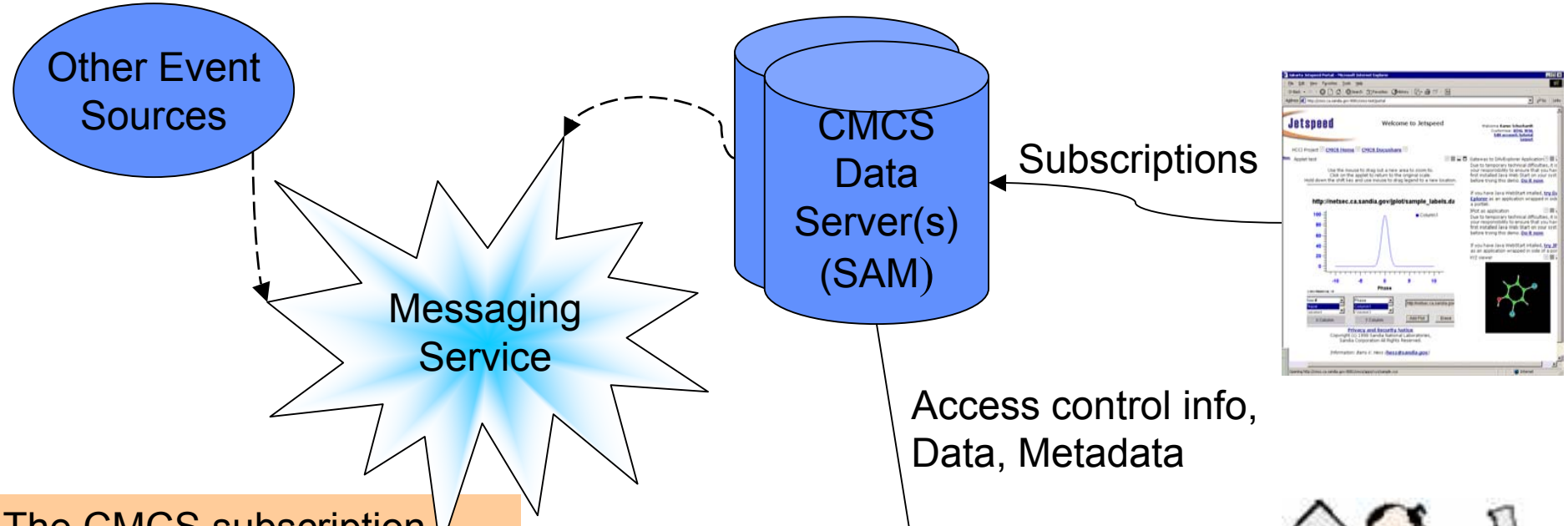
Subscription agent

Active Tables.

…

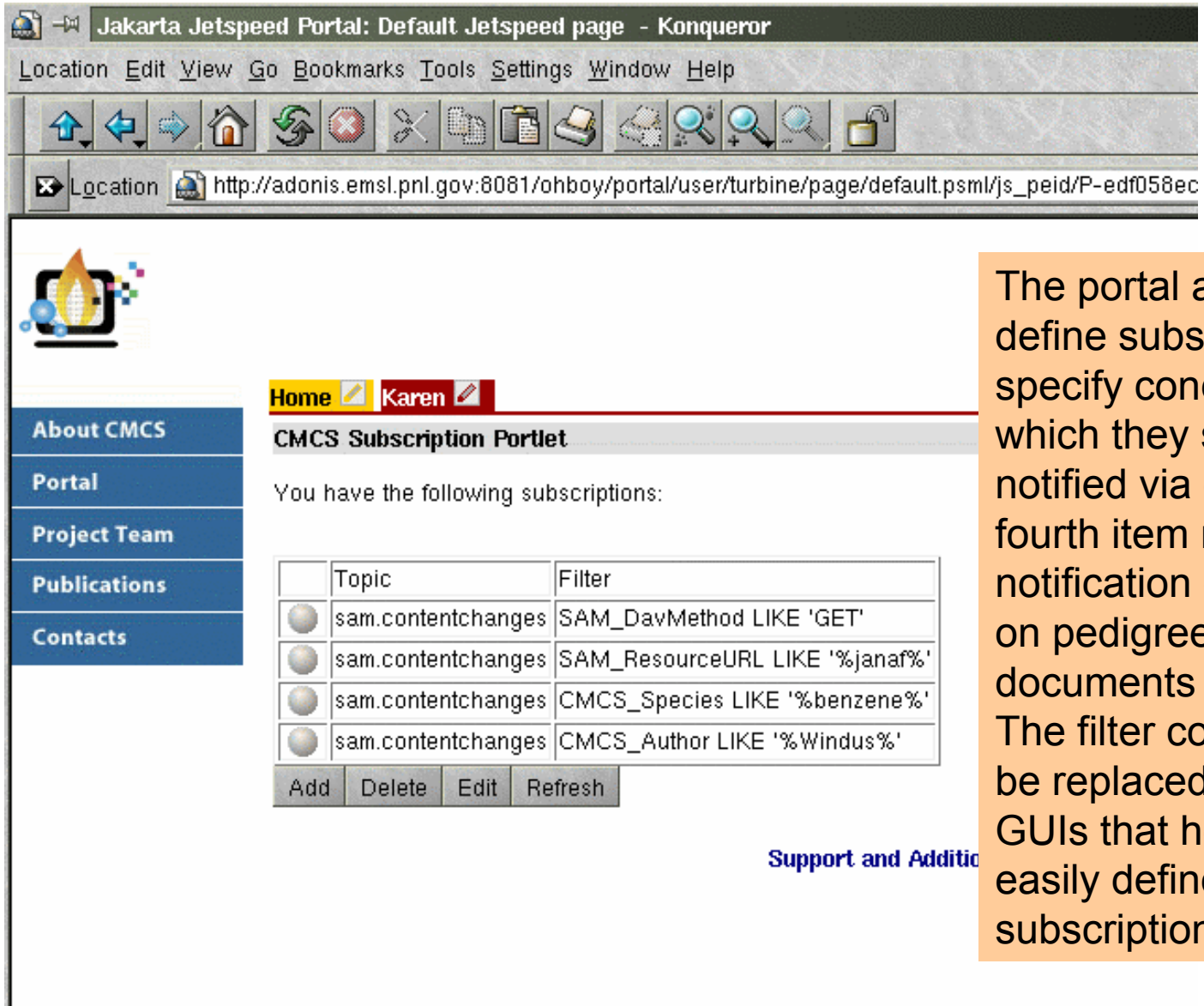
As users add/modify/remove data and metadata notifications are automatically generated. These notifications are available to both CMCS and custom built applications

Use Case – User Notifications



The CMCS subscription agent manages user subscription lists and applies complex filters to ensure users receive notification of events and data that are both relevant and accessible. The subscription lists are managed through the portal

Subscriptions in the Portal



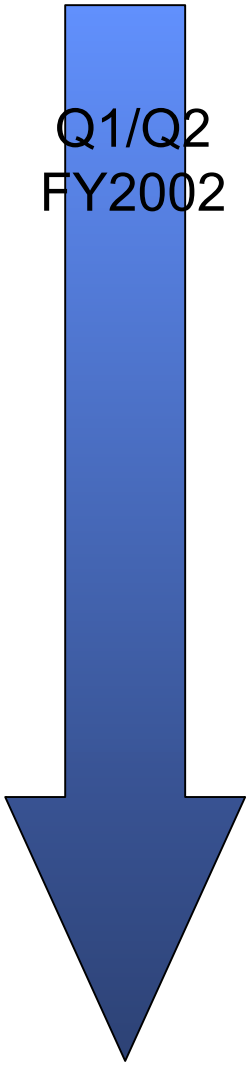
The portal allows users to define subscriptions that specify conditions under which they should be notified via email. The fourth item requests notification changes based on pedigree (all documents by Windus). The filter column will later be replaced by custom GUIs that help users easily define meaningful subscriptions.

Technology Overview

- Publish/Subscribe architecture
  - http://scidac.ca.sandia.gov/Get/File-409/Publish-Subcribe.ppt
- Lots of alternative solutions/technologies
  - Java Messaging System (JMS) (robust),
  - JXTA protocols (under development)
  - Xevents (grid events – high performance, high reliability) (under development)
  - Jabber (scalable, robust,publish/subscribe not well established)
- Initial decision – JMS (OpenJMS)
- Long term strategy - XML (language independence)

Notification Tasks

Q1/Q2
FY2002



- Assess messaging technology, make initial selection – (OpenJMS server now running at Sandia)
- Develop automated data store notifications (SAM) (messaging enabled server running at Sandia)
- Demonstrate executing legacy application based on data change notifications
- Investigate, track, participate in messaging related standards development (JXTA, XEvents…)

Notification Tasks

Q3/Q4
FY2002

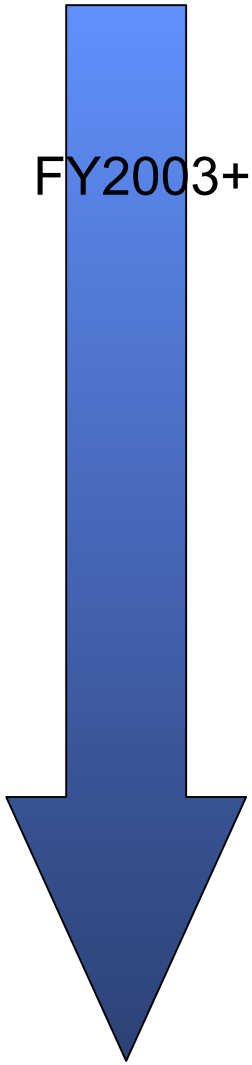
Develop initial end-user subscription agent with portal integration

Develop smart subscription agent

- Based on data and metadata content
- Checks access controls

- Develop a web service proxy to the message service to provide convenient integration with legacy apps (Fortran). Support sending messages and polling.
- Connect high performance data store for persistent messages.

Notification Tasks



- Develop dynamically configurable portlet subscription system (requires registration of message schemas)
- Add support for sending notifications to PDAs
- Investigate use of messaging for inter-portlet communications
- Develop logging/query capability
- Add authorization control over message content and distribution
- Migrate to protocol/XML-based service (XEvents, Jabber…)